WHAT IS CLAIMED IS:

1. A method for use in a fixed point arithmetic processing device having an input vector that contains one or more vector elements, and is an M bit integer, and a maximum permitted left shift (MLS) value for the input vector is less than or equal to the value of M - 2, the method for scaling all the vector elements based on the vector element with the largest magnitude, the method comprising:

sequentially searching each vector element to find a left shift value for scaling each vector element;

comparing the left shift values to determine a minimum left shift (NLS_MIN) for scaling the largest vector element;

employing the NLS_MIN value to determine whether the input vector is a non-zero input vector;

if so, regardless of whether the largest element of non-zero input vector has a positive or negative magnitude, offsetting the NLS_MIN value by the MLS value to obtain an actual number of left shifts (NLS) value for scaling the input vector;

determining whether the input vector is a zero input vector; and

if so, offsetting the NLS_MIN value by the MLS value to obtain the NLS value.

2. The method of claim 1 further comprising employing a pdmsb instruction for sequentially searching, and for comparing said left shift values.

3. A method, by a processing device, for scaling an M-bit integer input vector containing one or more vector elements, said method comprising:

receiving a maximum permitted left shift (MLS) value for the input vector, said MLS value being less than or equal to M - 2;

determining a minimum left shift (NLS_MIN) for scaling said vector element with the largest magnitude;

employing said NLS_MIN value to determine whether said input vector is a zero input vector, or a non-zero input vector irrespective of the positive or negative value of said non-zero input vector;

if a non-zero input vector is determined, offsetting said NLS_MIN value by said MLS value to obtain an actual number of left shifts (NLS) value for scaling said input vector; and

if a zero input vector is determined, offsetting said NLS_MIN value by said MLS value to obtain the NLS value.

4. The method of claim 3 wherein offsetting said NLS_MIN value for said zero input vector further comprises said NLS value being given by: MLS + 1.

5. The method of claim 3 wherein offsetting said NLS_MIN value for said non-zero input vector further comprises said NLS value given by:NLS = NLS_MIN + (MLS-(M-2)).

6. The method of claim 3 further comprising employing a pdmsb instruction for sequentially searching, and for comparing said left shift values.

7.. The method of claim 3 wherein employing said NLS_MIN value further comprises determining whether NLS_MIN = 31, if NLS_MIN $\neq$ 31, then the input vector is a non-zero input vector.

8. A processor operable from an M-bit instruction set where M is an integer, the processor comprising:

a memory unit for storing at least first instruction stream comprising M-bit instructions;

an execution unit operable to receive execution signals to execute the M-bit instructions;

a decode unit coupled to the memory unit and to the execution unit to receive and decode the first instruction stream from the memory unit to produce therefrom the execution signals, the execution signals for:

determining a minimum left shift (NLS_MIN) for scaling said vector element with the largest magnitude;

employing said NLS_MIN value to determine whether said input vector is a zero input vector, or a non-zero input vector by evaluating if NLS_MIN = 31;

if NLS_MIN $\neq$ 31, then the input vector is a non-zero input vector; and

determining an actual number of left shifts (NLS) for scaling the non-zero input vector.

9. The method of claim 8 wherein the execution signals is for receiving a maximum permitted shift (MLS) value for said input vector, said MLS value being less than or equal to M - 2.

10. The method of claim 9 wherein determining an actual number of left shifts (NLS) further comprises offsetting said NLS_MIN with the MLS value to obtain said NLS value.